IMAGE CLASSIFICATION

PROJECT DETAILS:

DATASET

Source:Accessed the CIFAR-10 dataset from keras library

Contents: A collection of 60,000 color images classified into 10 classes

Data Loading:

- Data is loaded using cifar10.load_data( ) function – loads both training and testing datasets

Preprocessing:

- Ensure all the images are in same format – size
- Split training data into training and validation
- Formed a dictionary mapping integer values to corresponding labels
- Transformed label_encoded values into one_hot encoded values
- Converted pixel values to float.
- Scaling of pixel values to a range of [0,1] for better training and faster convergence.

Model Architecture:

- **Convolutional Blocks**: The model is built with three convolutional blocks. Each block includes a Conv2D layer, BatchNormalization, and MaxPooling2D. Last block includes a dropout (0.3) layer.
- **Fully Connected Layers**: After flattening the convolutional layers' output, the model has two dense layers with 32 and 64 units. These layers are followed by BatchNormalization and Dropout (0.5) to improve performance and prevent overfitting.
- **Output Layer**: A Dense layer with 10 units and a softmax activation function for classification.

Model Training:

- Adam optimizer is used
- Compiled the model with categorical cross entropy loss function
- Trained for ten epochs with a batch size of 64

- Plotted training and validation accuracy to observe the model's performance throughout the training process.

Model Evaluation:

- After predicting ,the max probability value is obtained using argmax() function.
- Evaluated the model's accuracy by generating a classification report.
- Accuracy of the model is 73%.

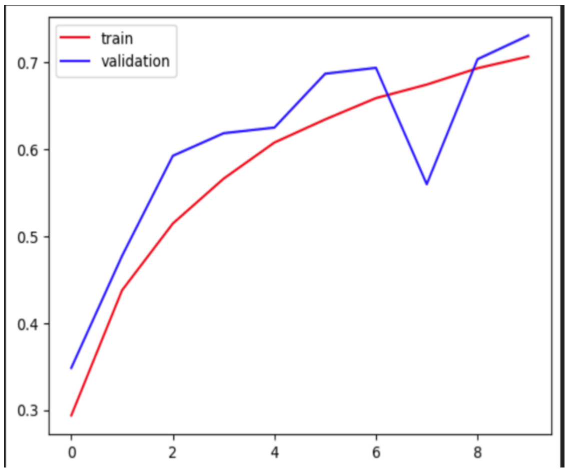
CHALLENGES& SOLUTIONS:

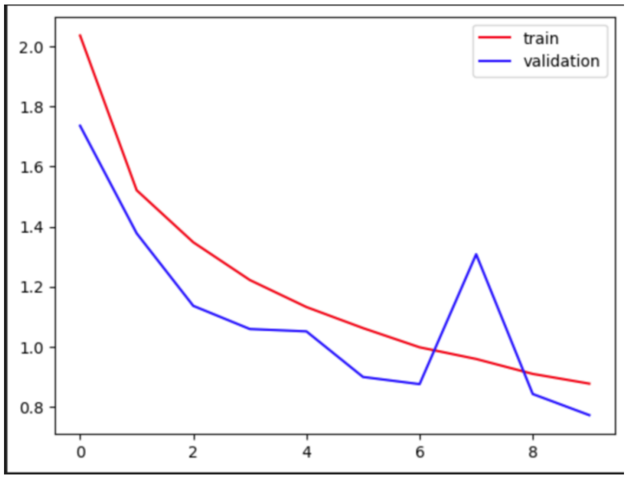
1. **Overfitting:** The model was overfitting due to complex structure
   Solution: regularized the model using Batch Normalisation and Dropout layers.
2. **Normalisation**: slow convergence of the model
   The model's training improved after normalizing the values to a range of [0,1]

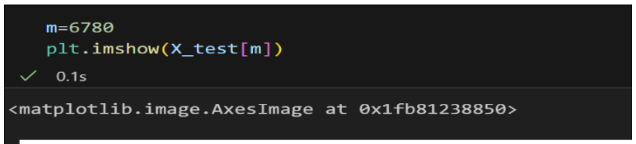
Plots- training and validation (x-axis-epochs)

a)accuracy plot

b)loss



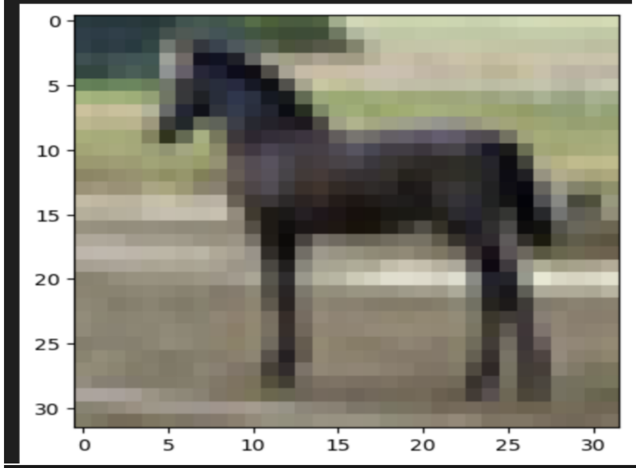


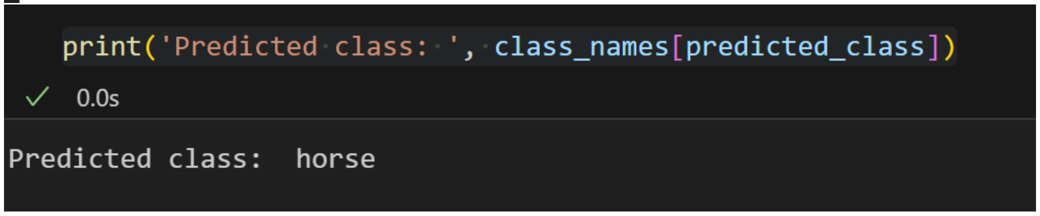


```
m=6780
plt.imshow(X_test[m])
```

✓ 0.1s

```
<matplotlib.image.AxesImage at 0x1fb81238850>
```





```
print('Predicted class: ', class_names[predicted_class])
```

✓ 0.0s

```
Predicted class:  horse
```